Understanding Cascading Menus - Jim Brossman

I have been using menus, sub menus and cascading menus for many years. One program has 7 menus, 35 sub menus and 5 of the sub menus have another 22 cascading menus. Recently I wanted to add a cascading menu for a sub menu and realized I really didn't understand what I was doing. Thanks to Alyce Watson's Companion for LB 4 and an included demo by Brent Thorn I was able to finally understand how to end up with what I wanted.

The first thing that wasn't obvious was that the menu id or number order was in the order the menus were entered in the program but starting at zero. So the first menu you enter has the id number of 0 and the next has the id of 1 etc. Sub menus under a menu also have a position number starting with 0 as we shall see later.

Cascading menus that will modify sub menus will also have an id number in the order that they are written into the program. If you look at the first three "menu" lines in the demo program, you will see they are labeled Menu #0 through Menu #2. Menu #0 and Menu #3 will have some of their sub menus modified to add cascading menus.

The next five menu lines will be the replacements of the sub menus to add the cascading menus to the sub menus. Note that they have a menu number from 3 to 7 because they are the 4th through 8th menu lines to appear in the program.

Finally there is one more menu line labeled Menu #8 which is the 9th menu line and it will not be modified. It make sense to me to put Menu lines #3 through #7 either at the end of the menu lines or as I did, just before the last menu lines that will not be modified which in this case is Menu # 8. They should be grouped together so they can be removed with a for next loop.

I won't go into the coding to modify and remove the cascading menus because that has been written by others. If you look at the program you will see a remarked 'wait statement just after the trapclose line. If you unremark the wait and run the program you will see that every menu line you have written is in the window in the order you have written them.

The lines following the wait statement are needed to get the handle of every menu line that will be changed or removed. The first line gets the handle of the window menu while the next lines get the handles of the different sub menus.

Next we must modify the sub menus to add the cascading menus and this is where it gets confusing. First we used the Menu # to retrieve the submenu handles. Now we are going to use the position number which is the position of the sub menu under the each menu starting with zero. So in the first line:

```
call MenuMod hnames, 0, heditnames, "&Edit Names"
```

"Edit Names" is the first sub menu under "Names" so its position is 0.
In the second line:

```
call MenuMod hnames, 2, hfinddup, "&Find Duplicate"
```

“Find Duplicate” is the third sub menu under “names” so its position is 2. Each sub menu under a menu item will start at position 0.
Next we must remove the cascading menus because they have now all been added to the original sub menus. If we put a wait after modifying the sub menus we would see the cascading menus # 3 through 7 are still at the top of the window in the window menu even though the sub menus have the cascading menus in them. They must be removed and the window menu must be drawn again.

Note that I used a for next loop to remove each menu 3 through 7 but, and this is important, they must be removed starting with the last menu # and working toward the first. If you remove menu #3 first, then menu # 4 becomes #3, menu #5 becomes 4 etc. You can also remove one line at a time but it is still easier to remove them backwards. Finally the window menu must be redrawn.

```
'Cascading Menu Demo - Jim Brossman

    nomainwin
    WindowWidth = 600 : WindowHeight = 300
    UpperLeftX=int((DisplayWidth-WindowWidth)/2)
    UpperLeftY=int((DisplayHeight-WindowHeight)/2)


'These are the menu items that will be modified to have cascading menu
s added.

'The label [anybranch] must be in the menu but does not need to be def
ined.
    'You can't leave it blank.
    menu #main, " &Names ", "Edit Names", [anybranch],
_          'Menu #0
        "&Print Names", [PrintNames],_
        "Find Duplicate",[anybranch]
    menu #main, " &Address ", "&Edit Address", [EditAdd]
         'Menu #1
    menu #main, " &Tools ", "Sub Menu 1",
_                      'Menu #2
     [SubMenu1],"Sub Menu 2", [SubMenu2],_
      "Sub Menu 3", [anybranch]

    'These will be the sub menus that modify the above menus and
    'add the cascading menus.
    menu #main, "&Edit Names","&Add",[Add],"&Change",[Chg],
_     'Menu #3
        "&Delete",[Del]
    menu #main, "&Find Duplicate", "&Names", [FindDupNames],_
```

```
'Menu #4
    "&Addresses", [FindAdds]
    menu #main, "&Sub Menu 1", "&Top", [Top], "&Middle", [Middle],
_     'Menu #5
    "&Bottom", [Bottom]
    menu #main, "&Sub Menu 2", "&First", [First], "&Second", [
Second],_ 'Menu #6
    "&Third", [Third]
    menu #main, "&Sub Menu 3", "&One", [One], "&Two", [Two],_
'Menu #7
    "T&hree", [Three]

    'These is another menu items that will not be modified.
    menu #main, " &Help ", "&Help File", [HelpFile],
_            'Menu #8
    "Help &About", [HelpAbout]

    open "Cascading Menu Demo" for window as #main
    print #main, "trapclose [Quit]"
    'wait  'Unremark this line to see all menus in the window.

    'Get handles of the main window, window menu and sub menus.
    hMain=HWnd(#main)'Handle of main window
    'Get the handle of the window menu into hMenu
    calldll #user32, "GetMenu",hMain as ulong,hMenu as ulong
    'Get the handles of the sub menus from hMenu using the Menu #
    'and put the handle into hnames, htools, heditnames etc.
    'We are going to modify menu #0 with menu #3 and #4
    'and menu #2 with menus #5, 6 and 7.
    calldll #user32, "GetSubMenu",hMenu as ulong,_
    0 as long,hnames as ulong  'Get handle of the Names menu #0
    calldll #user32, "GetSubMenu",hMenu as ulong,_
    2 as long,htools as ulong  'Get handle of the Tools menu #2
    calldll #user32, "GetSubMenu",hMenu as ulong,_
    3 as long,heditnames as ulong  'Get handle of Edit Names menu #3
    calldll #user32, "GetSubMenu",hMenu as ulong,_
    4 as long,hfinddup as ulong  'Get handle of Find Duplicate menu #4
    calldll #user32, "GetSubMenu",hMenu as ulong,_
    5 as long,hsubmenu1 as ulong  'Get handle of Sub Menu 1 menu #5
    calldll #user32, "GetSubMenu",hMenu as ulong,_
    6 as long,hsubmenu2 as ulong  'Get handle of Sub Menu 2 menu #6
    calldll #user32, "GetSubMenu",hMenu as ulong,_
    7 as long,hsubmenu3 as ulong  'Get handle of Sub Menu 3 menu #7

    'Modify the submenus to add the cascading menus.
```

```
'To modify a sub menu you need the handle of the menu item from the ab
ove lines,
    'the the sub menu position under the menu item,
    'the handle of the sub menu to make that will make the change,
    'and the new name of the sub menu.
    call MenuMod hnames, 0, heditnames, "&Edit Names"
    call MenuMod hnames, 2, hfinddup, "&Find Duplicate"
    call MenuMod htools, 0, hsubmenu1, "Sub Menu &1"
    call MenuMod htools, 1, hsubmenu2, "Sub Menu &2"
    call MenuMod htools, 2, hsubmenu3, "Sub Menu &3"


'Remove all the cascading sub menus now that they have replaced the
    'original ones because they still remain in the window menu.
    'Remove in reverse order or numbers will be wrong because if you
    'remove 3 first, #4 becomes #3, #5 becomes #4 etc.
    for x=7 to 3 step -1
        call MenuRemove hMenu, x
    next

    'You must redraw the menu bar for all the changes to take effect.
    calldll #user32, "DrawMenuBar", hMain as ulong, result as long
    wait


    'Sub to modify the sub menus and add the cascading menu.
    sub MenuMod hmainmenu, position, hsubmenu, pointer$
        flags = _MF_BYPOSITION Or _MF_POPUP Or _MF_STRING
        calldll #user32, "ModifyMenuA",_
            hmainmenu as ulong,_  'Handle of the menu to be modified
            position as long,_
'position of the sub menu under the menu
            flags as long,_ ' change using the position number
            hsubmenu as ulong,_ 'hendle of sub menu to change
            pointer$ as ptr,_ 'new name of the sub menu
            result as long
    end sub

    'Sub to remove cascading menus no longer needed.
    sub MenuRemove hmainmenu, position
        calldll #user32, "RemoveMenu",hmainmenu as ulong,_
'handle of window menu
        position as long,_MF_BYPOSITION as long,_
'Menu # of original cascading menu.
        result as long
    end sub
```

```
'These branches are here so the program doesn't crash if
'you select a menu item.
[Add] a$="Add": gosub [Notify]:wait
[Chg] a$="Change": gosub [Notify]:wait
[Del] a$="Delete": gosub [Notify]:wait
[PrintNames] a$="Print Names": gosub [Notify]:wait
[EditAdd] a$="Edit Addresses": gosub [Notify]:wait
[FindDupNames] a$="Find Duplicate Names": gosub [Notify]:wait
[FindAdds] a$="Find Duplicate Adresses": gosub [Notify]:wait
[Top] a$="Top": gosub [Notify]:wait
[Middle] a$="Middle": gosub [Notify]:wait
[Bottom] a$="Bottom": gosub [Notify]:wait
[First] a$="First": gosub [Notify]:wait
[Second] a$="Second": gosub [Notify]:wait
[Third] a$="Third": gosub [Notify]:wait
[One] a$="One": gosub [Notify]:wait
[Two] a$="Two": gosub [Notify]:wait
[Three] a$="Three": gosub [Notify]:wait
[HelpFile] a$= "Help File": gosub [Notify]:wait
[HelpAbout] a$= "Help Aboute": gosub [Notify]:wait

[Quit]
close #main
end

[Notify]
Notice "You have selected "+a$:return
```